REMARKS

Favorable reconsideration of this application, as presently amended and in light of the following discussion, is respectfully requested.

Claims 16-18 and 20-32 are pending, with claims 16 and 20-31 amended, and claim 32 added by the present amendment. Claims 16, 23, 29 and 31 are independent.

In the Official Action, the IDS of September 27, 2004 was objected to; claims 16, 23 and 27-28 were objected to; claim 31 was rejected under 35 U.S.C. § 103(a) as being unpatentable over Nikhil (U.S. Patent No. 5,499,349); claims 16-18, 20-25 and 28-30 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Nation (U.S. Patent No. 6,233,599) in view of Nikhil; and claims 26-27 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Nation in view of Nikhil and Meyer (U.S. Patent. No. 7,640,315).

Claims 16 and 20-31 are amended in response to the objections thereof and to more clearly describe and distinctly claim Applicant's invention. New claim 32 corresponds to previously cancelled claim 19, albeit amended to maintain antecedent basis. Support for this amendment is found in Applicant's originally filed specification.[1] No new matter is added.

Briefly recapitulating, amended claim 16 is directed to:

> An N+1 parallel program (P-P) module based on a single machine environment, comprising:
>
> an N+1 P-P branch programs module, where N is greater or equal to 1, which is run by operating the N+1 P-P branch programs which have an object code independent structures, in the way of time division, for making a transmission and consistency of P-P data in the P-P branch programs under the supports of three classes of sequence-net instructions for reading P-P data, writing P-P data, and making P-P data consistency; and

[1] Specification, at least page 13 (last paragraph), page 15 (last paragraph) and page 16.

a managing program module, for supporting a suspension status, a ready status, and a running status of the P-P branch programs in response to information from the P-P branch programs,

wherein the classes of the sequence-net instructions for reading P-P data and writing P-P data are only executed by the 1st to Nth P-P branch programs, and the class of the sequence-net instruction for making P-P data consistency is only executed by the N+1th P-P branch program.

The Present Invention

The present invention relates to parallel processing (P-P). As stated in the Background of the present application, in the prior art, a so-called "parallel flow chart" was considered as N flows and interconnections of data, which can be referred to as "N P-P". Later on, a so called "sequence-net-computer" was proposed and referred to as "N+1 P-P", in which an N+1th branch program was newly added for the purpose of data consistency.

One important point is that both N P-P and N+1 P-P are parallel processing structures for **multi-machine environment**, i.e., each process out of the "N" or "N+1" is executed by a separate machine. In the "N+1 P-P", the N+1th branch program is responsible for transmitting and synchronizing the data among the other N programs, which is realized by distributed tokens, in particular, by three new instructions: write data instruction, read data instruction and P-P data consistency instruction. The N+1th branch program is only used for driving the P-P data sequence without any normal numerical calculation although it is at the same level as the other N branch programs in view of the structure.

The newly introduced N+1th branch program not only can harmonize the data among the other N branch programs, but also can enable the whole P-P structure to easily support a nesting function, since it can also harmonize the context between a parent P-P and a child P-P. This is one of the prominent advantages of N+1 P-P over N P-P.

However, in a **single-machine environment**, the P-P branch programs have to be merged into one sequence and executed in a time division manner. Fig. 1B in the present application shows a direct combination of N+1 P-P, resulting in an inseparable serial program, which loses the original structure of the P-P and thus the data sequence is fixed and the nesting function is not supported anymore.

In view of the above problem, the present invention includes a new scheme for transplanting the multi-machine P-P into a single-machine environment as defined in the independent claims. In the present invention, a logic parallel structure of the N+1 P-P is still maintained, the $N+1^{th}$ branch program is still responsible for the P-P data sequence for the other N branch programs, and the three instructions (i.e., the distributed token in the multi-machine environment) for data writing, data reading and P-P data consistency could be replaced with three subroutines.

Through such a transplanting scheme, the logic parallel structure of the N+1 P-P in the single machine environment can overcome the problem of fixed data sequence of a serial combination in the prior art. Further, with the $N+1^{th}$ P-P branch program dealing with the data sequence, nesting of one or more child P-Ps can be supported. In addition, the structure of the present invention can be well utilized in the Object Oriented (OO) technology, i.e., the whole structure can be packaged as an object. Please see pages 29-30 (points "1" to "6") in the original description for details of the above and other advantages of the present invention.

The Cited Prior Art Documents

Nikhil describes a multithreaded parallel data processing system including at least one processing element for processing multiple threads of computation. It is well known in the art

that multithread technology was born for a single machine environment, and thus it, by nature, has nothing to do with P-P. The processing element 10 of Nikhil is capable of processing a plurality of threads of computation independently (Nikhil: column 3, line 21-22), preferably by executing a reduced instruction set (Nikhil: column 5, line 13-14) including the new control instructions such as fork, join, and start instructions (Nikhil: column 7, line 50-52). Therefore, the join instruction cited by the Official Action joins together two threads, which means the processing element 10 joins together the two threads by executing a join instruction. That is, the sequence of multiple threads is controlled by the processing element 10 which is equivalent to a parent controlling module. Therefore, Nikhil does not disclose or suggest Applicant's claimed feature of "***wherein the classes of the sequence-net instructions for reading P-P data and writing P-P data are only executed by the 1st to Nth P-P branch programs, and the class of the sequence-net instruction for making P-P data consistency is only executed by the N+1th P-P branch program***." Specifically, any joint instruction in Nikhil is executed by the both threads to be combined rather than by another independent thread. Specifically, with Applicant's amendments, it is now clear that the 1st to Nth P-P branch programs execute normal calculation and thus need to read and write P-P data, while the N+1th P-P branch program is responsible for the P-P data consistency without the need to read and write P-P data. Such kind of a parallel structure with an additional branch for executing the P-P data consistency of the other branches is not disclosed in the multithread structure of Nikhil.

Nation describes an apparatus and method for performing multithreaded operations which partitions an existing processor register set into register subsets, allocates the register subsets to a plurality of threads such that each thread has an associated register subset which stores that thread's resources (Nation: column 2, line 63 to column 3, line 4). Figure 1a of Nation shows a

plurality of thread context planes 60 is controlled by the thread controller 50. That is, the sequence of the thread context is controlled by the thread controller which is equivalent to a parent controlling module. Therefore, Nation also does not disclose or suggest Applicant's claimed feature of "***wherein the classes of the sequence-net instructions for reading P-P data and writing P-P data are only executed by the 1st to Nth P-P branch programs, and the class of the sequence-net instruction for making P-P data consistency is only executed by the N+1th P-P branch program***."

Applicant has considered Meyer and submits Meyer does not cure the deficiencies of Nikhil and Nation. Applicant further submits that independent claims 23, 29 and 31 patentably define over the applied references for reasons similar to those identified above relative to amended independent claim 16. As none of the cited art, individually or in combination, disclose or suggest at least the above-noted features of independent claims 16, 23, 29 and 31, Applicant submits the inventions defined by claims 16, 23, 29 and 31, and all claims depending therefrom, are not rendered obvious by the asserted references for at least the reasons stated above.

MPEP 2141 notes that prior art is not limited just to the references being applied, but includes the understanding of one of ordinary skill in the art. MPEP 2141 further notes that the prior art reference (or references when combined) need not teach or suggest all the claim limitations. However, an obviousness-type rejection must explain why the difference(s) between the prior art and the claimed invention would have been obvious to one of ordinary skill in the art. MPEP 2141 goes on to list exemplary rationales that may support a conclusion of obviousness. However, Applicant submits that the Official Action and the applied references present no objective evidence that would support an obviousness-type rejection of Applicant's amended claims based on one of these exemplary rationales.

Turning now to dependent claim 32 (corresponding to features of previously pending claim 19 that have been deleted from the independent claims), Applicant does not agree that Fig. 9 (element 76) and column 8, lines 8-31 of Nikhil disclose or suggest an N+1th P-P branch program that executes a P-P data sequence, which is represented by a data consistency operation. The joint instruction in Nikhil combines two independent threads into a single thread. The part of the description cited by the Official Action does not mention anything concerning the processing of the **data** of the two threads to be combined. The Official Action alleges that the joint instruction of Nikhil reads upon the data consistency operation, since the joint instruction of Nikhil allows for synchronization between the two threads by having the first thread wait until the second executes the joint instruction and thus allows for consistent results. However, what can be read from Nikhil is only the synchronization of the timing of the two threads rather than the timing of the data. The Official Action's interpretation of the joint instruction is indeed made after knowing the present invention, and such an assertion using hindsight is of course not convincing. Therefore, Nikhil does not disclose "the N+1th P-P branch program executes a P-P data sequence, which is represented by a data consistency operation." Thus, for independent reasons, new claim 32 patentably defines over the applied references.

CONCLUSION

Should there be any outstanding matters that need to be resolved in the present application, the Examiner is respectfully requested to contact Michael E. Monaco, Reg. No. 52,041, at the telephone number of the undersigned below, to conduct an interview in an effort to expedite prosecution in connection with the present application.

If necessary, the Commissioner is hereby authorized in this, concurrent, and future replies to charge payment or credit any overpayment to Deposit Account No. 02-2448 for any additional fees required under 37.C.F.R. §§ 1.16 or 1.147; particularly, extension of time fees.

Dated: OCT 08 2010

Respectfully submitted,

By _____

James M. Slattery
Registration No.: 28380
BIRCH, STEWART, KOLASCH & BIRCH, LLP
8110 Gatehouse Road, Suite 100 East
P.O. Box 747
Falls Church, VA 22040-0747
703-205-8000

Michael E. Monaco
Registration No. 52,041